

Dynamic Web-Service Composition: eBusiness Just For You!

Steffen Higel, Dave Lewis, Vincent Wade

Department of Computer Science, Trinity College Dublin

The Challenges of the Information Society/ Information technology and Systems

Working Paper

ABSTRACT

As businesses increasingly strive to offer 'service portals' on the web, their business processes are being encapsulated within web-services or offered via web services. Initially, this has been particularly focused in the business to customer area, for example on-line brokerage services, information search services, online travel agents and so forth. However, in recent years the take-up of web services to support the business to business market has also increased through their appearance across the value chain of telecoms providers[Lofthouse04] and the emergence of value added service providers (such as content providers and retailers). In supporting these B2B and B2C business processes, a variety of technologies such as workflow automation, scripting languages or individual applications supported via application service providers have been employed. Market forces as well as operating constraints can cause the need for dynamic changes to both the business models and services that an organisation offers. However all of the technical solutions listed earlier suffer from a lack of dynamic adaptability. Such solutions incur considerable effort in process re-engineering and customisation. More lightweight solutions to service composition are now being sought to allow organisations greater flexibility and more rapid process adaptivity. Various industry and standards bodies have reacted by attempting to specify greater flexibility in Web Service composition and integration. For example, W3C as part of its Web Service Activity has sought to define requirements and techniques to support dynamic service choreography¹; OASIS is standardising a scripting approach for Web Services called

¹ <http://www.w3.org/2002/ws/chor/>

Business Process Execution Language for Web Services². This paper explores an approach to near run-time service composition for customising web-service behaviour dynamically, with a focus on the reuse of existing compositions.

INTRODUCTION

Presently, the typical business to consumer model on the Internet consists of a human selecting and buying products using a web browser. Emerging from this current relationship is the notion that the process of interacting with the various components of the purchasing process (e.g. choosing a product, buying it, paying for it and choosing a carrier to ship it) can be rigidly defined and standardised, such that a suitable piece of software could begin to help a user in the process of finding a combination of merchants which best suits their requirements and preferences. This rigid standardisation forms the basis of what are known as Web Services. Web Services rapidly displayed the ability to address deficiencies in other areas of computer aided communications and resource management [Martin-Flatin and Doffoel, 2003]. They can provide bridges between computer programmers working in different languages on different platforms and provide a means for managing a huge range of devices, from light switches to telephone switches. However, it is only when many of these Web Services are woven together into a larger piece of software does their true benefit become apparent [Orriens et al, 2003]. Traditionally, tailored software development has consisted of a user approaching someone with sufficient skill to piece together something which fulfils their needs. However, with a sufficiently detailed description of each Web Service, larger, more useful composed services can be built easily as a user needs them.

This paper will show the reader how our proposed dynamic Web Service Composition system works and why it will provide a novel method for improving the reliability and adaptability of

² <http://www-106.ibm.com/developerworks/library/ws-bpel/>

compositions.

DRIVING FACTORS

It is self evident that organisations which have the capability to rapidly change the nature of their customer and business to business relationships increase their chances of growing during the good times and surviving during the bad ones. We believe that web services offer businesses of all sizes and natures the opportunity to quickly offer relevant parts of their business over the Internet. When the methods for description and communication are standardised, it becomes possible to rapidly change the way the services work "behind the scenes" without needing to inform software clients of any new configuration requirements. When we define a chain of these services to interact with one another, we can trivially replace one service with another which provides the same functionality but with higher reliability or at a more competitive price.

When we begin to use software to generate compositions of services dynamically, we are faced with the harsh reality that the process is both error prone and computationally expensive.

Therefore, compositions that are known to work well should have a certain amount of value placed upon them. Typically, a computer scientist would label these previously used verified compositions as 'patterns'. Indeed, it is entirely feasible that revenue opportunities will emerge that focus purely on two areas core to our vision of dynamic web service composition: brokering, whereby a trusted third party searches for web services which are best suited to a client's needs, and pattern resale, whereby the development of high quality patterns can be licensed for use.

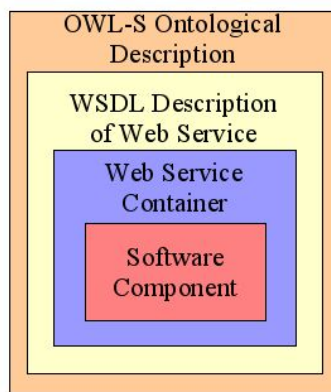
CURRENT APPROACHES

In our belief, there are three core requirements to facilitate research into truly flexible and reliable automated service composition; appropriate semantic standards, the support of existing areas of mature research, in particular those of planning and workflow and finally robust tools for

creating, interacting with and monitoring compositions. We will examine the first two, the final one itself being an enormous task and is beyond the scope of this paper.

Semantic Standards

The area of semantic standards is one of ongoing research. There are many proposed methods and levels to describe services, and indeed to allow them to intercommunicate. At the heart of this problem is finding a suitable balance between imposing structure on service descriptions (to make them as machine interpretable as possible) without impeding this description from being as



verbose as they possibly can (because ultimately the more verbose the service description, the more a piece of software can understand what it does and how to use it correctly).

Much like the OSI Model of a Network Stack, most research is tending towards a multi-layered approach to service description. At the lower levels, services are described purely functionally, that is they are described in terms of what goes in one end and what comes

out of the other. A service for manipulating access control on a firewall might take four inputs (the IP address whose access is being altered, the alteration to be made and the username and passwords of the person making the alteration) and would produce one output (either a confirmation or rejection message). At present, the most popular means of expressing this is the Web Service Description Language (WSDL), an XML based language. A programmer could write a program in about 5 lines of code to translate a supplied string from English into French using the WSDL files published by the Babelfish online translation system. Without a standard like WSDL, the task of writing software to interact directly with the Babelfish service would be far more involved.

While WSDL offers sufficient semantics to write software which interacts directly with web

services, it lacks any real form of information for reasoning on not just what the inputs and outputs of a service are, but indeed what those inputs and outputs actually mean. For artificial intelligence techniques to be applied successfully to the task of automatically building service compositions, we really need a method for describing objects not just using words, but also in terms of interrelated concepts. A computer cannot understand the concept of a book without external help [Sirin and Parsia, 2004]. In eBusiness, an interaction with a web service might result in a book being shipped to the customer. We need a way of binding the concept of a book to a physical object which needs to be shipped through the post, so that a computer can reason on the correct series of events that are required to make this happen. These inter-relational concepts are called ontologies, with the DARPA Agent Markup Language (DAML) maturing into the specification upon which we will base our research. An extension of DAML, the Ontology Web Language for Services (OWL-S), seeks to "supply Web service providers with a core set of markup language constructs for describing the properties and capabilities of their Web services in unambiguous, computer-interpretable form."³

The Influences of Workflow and Planning

Workflow, defined in [Joosten, 1994] as "The set of relationships between all the activities in a project, from start to finish. Activities are related by different types of trigger relation. Activities may be triggered by external events or by other activities." It provides a formal method for expressing the tasks that need to be completed in order to fulfil some goal, while striving to ensure that the resources used to do this are used as efficiently as possible. The parallels with web service composition are clear and much work has gone into defining the common problems, most notably in [van der Aalst et al, 2003].

³ As defined in "OWL-S: Semantic Markup for Web Services", a technical overview published by the The OWL Services Coalition

Artificial Intelligence Planning seeks to represent a relevant part of the world in terms of various states and possible changes that can be made to those states. This rigid approach to world representation allows the usual suite of A.I techniques to be applied to a huge range of problems, automatic web-service composition included. In Web Service Composition, we can view the execution of a service as altering a set of resources or changing the information that a user has available to them. A branch of Planning known as Situational Calculus classifies the functional properties of a web service as Inputs, Outputs (states of knowledge of the user) and Preconditions and Effects (world states). A great amount of research in these areas has quickly advanced that of service composition.

PATTERNS: COST-EFFECTIVE AND EFFICIENT

While Planning offers us the ability to automate large portions of the service composition process, finding a solution can be a problem of arbitrary complexity. Likewise, errors caused by semantic gaps during the planning process could result in nonsensical compositions. For this reason, we strongly advocate the reuse of existing, verified compositions. By negating the need to calculate significant portions of a composition, we also gain a significant amount of confidence in the validity in the composition. We trade off an amount of flexibility in doing so, so an exploration of this approach will need to investigate how to minimise this reduction.

A pattern is most eloquently defined in [Alexander, 1979] as "a three-part rule, which expresses a relation between a certain context, a problem and a solution". To clarify, in this case context covers the domain of the composition of web services, the problem is defined on a case by case basis, that is to utilise an appropriate number of software services such that they change the state of the world to a given set of requirements and the solution (at least at this stage in the discussion) is a description of a set of services to be used in a given order.

Overview of Proposed Approach

Superficially, our approach is simple. When an existing composition, created by a human or an appropriate algorithm is used to achieve some goal, part of or all of the composition is extracted, appropriately abstracted and then stored in an accessible repository. When a new composition is being built by our system, the HTN planner will first attempt to resolve the gaps through the reuse of these patterns. The patterns can be described in precisely the same manner as web-services (see Figure 1), purely in terms of their inputs, outputs, required world-states and resultant world-states.

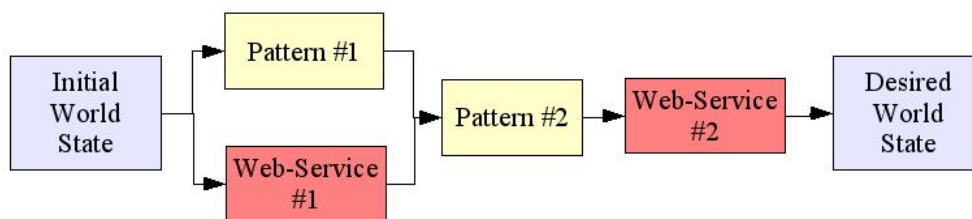


Figure 1 How patterns and services can co-exist within a composition

Problems Posed

Most significantly, representation of a previous composition presents a number of questions: Do we refer specifically to the services used inside this composition, or do we leave a more vague reference to their functionality, in case they are unavailable at some point in the future? When we personalise a composed service for a user, but want to save that composed service as a pattern, do we even attempt to express the non-functional properties of it?

Answering the first question, we do not feel that we gain much by referring to specific services in a composition. Some small computational savings are not worth the reduction in flexibility. It is more appropriate in this instance not to refer to each material element of the composition, but rather to use a general reference to its functionality. The implication of this is that we will have to resolve all of these functional descriptions to available services at composition time.

Experimentation alone will offer an answer to the second question. It is difficult to predict how

much user information is actually useful to the adaptativity process (though we could probably create an almost infinite list of criteria that might be useful). Previous research would suggest that a little bit of information can go an incredibly long way [Strachan et al, 1997], which will lead us towards a suitable technique for classifying a pattern's suitability.

Elaboration of the Pattern Driven Approach

Assuming we have extracted a large pool of verified, useful patterns and stored them in a suitable repository, we are presented with the issue of how to best put these patterns to use in future compositions. A general approach for using planning systems for web service composition has been proposed in numerous other papers [Wu et al, 2003], as has an overview of a pattern driven approach [Tut and Edmond, 2002], so an in depth discussion of the issues will not be necessary here. However, they do not take into account our core aims, that is, to maintain as much flexibility as possible and to use this to empower adaptivity.

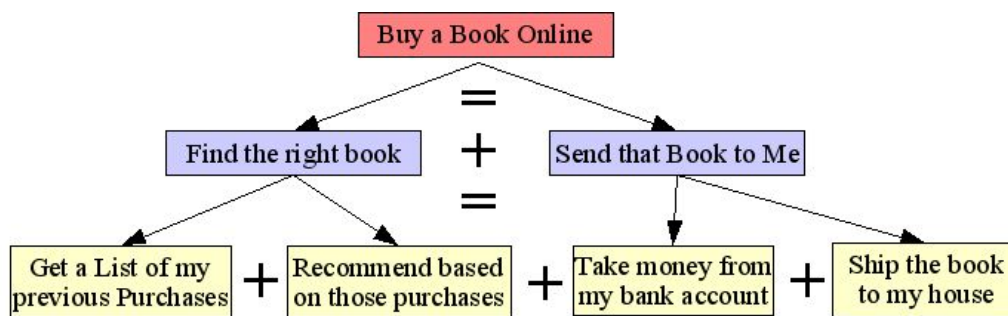


Figure 2A high level task being broken down into subtasks

As the planner begins to break down a high-level task into sub-tasks (as shown in Figure 2), we can very quickly bridge the emerging gaps by checking to see if we have a pattern which is already to do so. Because we haven't tied ourselves down to particular individual services, as discussed previously, we haven't lost as much flexibility as we might have. We have lost some structural flexibility, where we might have been able to adapt the order in which services are invoked to match some user preference (for example, the user might be willing to trade some

reliability or speed against cost). If an audit of the choice of pattern proves it to be highly unsuitable, we will allow the planner to bridge the gap itself, in the hope that it can either find a better pattern (likely) or generate a better composition from scratch (not so likely).

As we iterate further into the process, we will start to map the tasks onto actual instances of available services (for instance, we may work out that we need a service which will recommend a book to a person of a given set of interests). It is here that we feel that a large amount of adaptivity will actually take place, assuming we have a sufficiently large pool of services, as has been demonstrated by our research in the area of adaptive eLearning [Conlan et al, 2002].

FUTURE WORK

At present, we are focussed on experimentally verifying our ideas outlined in the previous section. This will consist of building a modular test bed (built around web services, for ease of inter-communication), allowing us to quickly answer the following open questions:

- Is a hierarchical task network based planner the most appropriate technology on which to base our research?
- What is the most appropriate way (if any) of describing the non-functional properties of a pattern?
- What user information significantly enhances our ability to create adaptive compositions?

The testbed will consist of a planner, which will need to have some custom developed proxies placed between it and the actual data with which it is working, depending on the original intended purpose of the selected planner.

The planner will be fed the information it needs and will be configured to produce a snapshot of its current work for every iteration it completes over the requested task. These snapshots will be analysed by a pattern matcher and then be fed into the planner. The analyses of the snapshot

might result in a pattern being rated as providing a reasonable but far from perfect match. In this case, we might fork the analysis, one path consisting of the in-progress composition with the pattern and one without. Once this framework is in place, we can begin to understand how best to rate the usefulness of a pattern in a given context.

While this method obviously lends itself well to the eBusiness domain, it is primarily being researched as part of the ubiquitous computing collaborative project, M-Zones. Integration with ongoing research into Context Management, Policy driven Security, quality of service requirements and ambient user interfaces are all part of this work. Classifying each of the above factors as simple functional inputs would allow the system to accept them using a single interface. However a discussion of this is beyond the scope of this paper.

CONCLUSIONS

We believe that dynamic web service composition will emerge as a new way of empowering businesses to provide their services in a market which rewards, above anything else, reliability and honesty and will allow users to interact with the businesses they perceive to be providing a service most suited to their needs. It is already quite easy to write software to bind together web services, but it is an easily automated task, provided we have appropriate semantic support.

Through the use of the area of artificial intelligence known as planning, we can automate not just what services are bound together, but also begin to generate complete interactions from scratch knowing only a few simple facts about the user and what their intentions are.

This paper has outlined a novel approach to personalised web service composition, which through the use of patterns intends to improve both the efficiency and the effectiveness of the composition process. We have discussed the established and important work being put into well defined semantic standards and the massive amount of useful research into Workflow. Our main

concern is maintaining a high degree of flexibility to facilitate user-centric adaptivity, which could very quickly be lost by pursuing a pattern driven approach. We therefore proposed that the easiest way to keep a composition pattern reusable is to always work on the assumption that the functionality offered by a service will be available (or can be fashioned together from other services) but that the service itself won't. Also while building new compositions using existing patterns, it is important never to assume that the first matched pattern is an absolute (we shouldn't be afraid of digging a little further in case we find something more appropriate). It is this problem that exposes one major question: how do we quantify the appropriateness of a structure? We then gave an overview of our plans to investigate the potential of these ideas.

REFERENCES

- [Lofthouse et al, 2004] **Parlay X Web Services** - H. Lofthouse, M. J. Yates, R. Stretch, January 2004, BT Technology Journal, Volume 22 Issue 1
- [Flatin and Doffoel, 2003] **Web Services for Integrated Management a Case Study** - J.P. Martin-Flatin and P.A. Doffoel, Technical Report DataTAG-2003-1 FP5/IST DataTAG Project, 22 November 2003
- [Orriens et al, 2003] **A Framework For Business Rule Driven Web Service Composition** - Bart Orriens, Jian Yang, Mike P. Papazoglou, 4th VLDB Workshop on Technologies for E-Services, 2003
- [Sirin and Parsia, 2003] **Planning for Semantic Web Services** - Evren Sirin and Bijan Parsia, Submitted to 3rd International Semantic Web Conference, 2003
- [Joosten et al, 1994] **An empirical study about the practice of workflow management** - Stef Joosten et.al., WA-12 report, 1994
- [van der Aalst, 2003] **Web Service Composition: Old Wine in New Bottles?** - W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede. Proceeding of the 29th EUROMICRO Conference: New Waves in System Architecture, pages 298-305. IEEE Computer Society, Los Alamitos, CA, 2003
- [Alexander, 1979] **The Timeless way of Building** - Alexander, C., Oxford University Press, New York, 1979
- [Strachan et al, 1997] **Pragmatic User Modelling in a Commercial Software System** - Linda Strachan, John Anderson, Murray Sneesby, and Mark Evans, Proceedings of the Sixth International Conference on User Modelling, 1997
- [Wu et al, 2003] **Automatic Web Services Composition Using SHOP2** - Dan Wu , Evren Sirin, James Hendler, Dana Nau, Bijan Parsia, Proceedings of The Twelfth International World Wide Web Conference, Budapest, Hungary, 2003
- [Tut and Edmond, 2002] **The Use of Patterns in Service Composition** – Moe Thandar Tut, David Edmond, Revised Papers from the International Workshop on Web Services, E-Business, and the Semantic Web, 2002
- [Conlan et al, 2002] **Multi-Model, Metadata Driven Approach to Adaptive Hypermedia Services for Personalized eLearning** - Conlan, O.; Wade, V.; Bruen, C.; Gargan, M. Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga, Spain, May 2002